**📘 Queue – Data Structure & Algorithm (Python)**

**🔹 What is a Queue?**

A **queue** is a linear data structure that follows the:

**FIFO** principle → **First In, First Out**

The first element added is the first one to be removed.

**🔹 Real-Life Examples:**

* Waiting line at a ticket counter
* Printer job queue
* Task scheduling
* Call center queue system

**🔹 Basic Operations:**

| **Operation** | **Description** |
| --- | --- |
| enqueue() | Add an item to the rear |
| dequeue() | Remove an item from the front |
| peek() | Get the front item without removing |
| is\_empty() | Check if the queue is empty |
| size() | Get the number of items |

**🔹 Visual Example**

FRONT → [10, 20, 30] → REAR

↑ ↑ ↑

dequeued first

**🔹 Queue Using Python List**

queue = []

# Enqueue

queue.append(10)

queue.append(20)

# Dequeue

front = queue.pop(0)

# Peek

print(queue[0])

# Check if empty

print(len(queue) == 0)

⚠️ Not efficient for large queues (because pop(0) is O(n))

**🔹 Efficient Queue Using collections.deque**

from collections import deque

queue = deque()

# Enqueue

queue.append(10)

queue.append(20)

# Dequeue

print(queue.popleft())

# Peek

print(queue[0])

✅ deque gives O(1) time complexity for append and popleft.

**🔹 Custom Queue Class (Using Deque)**

from collections import deque

class Queue:

def \_\_init\_\_(self):

self.queue = deque()

def enqueue(self, data):

self.queue.append(data)

def dequeue(self):

if not self.is\_empty():

return self.queue.popleft()

return "Queue Underflow"

def peek(self):

if not self.is\_empty():

return self.queue[0]

return "Queue is empty"

def is\_empty(self):

return len(self.queue) == 0

def size(self):

return len(self.queue)

def display(self):

print("Queue (front → rear):", list(self.queue))

**🔹 Example Usage**

q = Queue()

q.enqueue(10)

q.enqueue(20)

q.enqueue(30)

q.display() # Output: [10, 20, 30]

print(q.dequeue()) # Output: 10

print(q.peek()) # Output: 20

**🔹 Time Complexity**

| **Operation** | **Time Complexity (Deque)** |
| --- | --- |
| Enqueue | O(1) |
| Dequeue | O(1) |
| Peek | O(1) |
| is\_empty | O(1) |

**🔹 Applications of Queue**

✅ Used in:

* CPU/Job Scheduling
* Breadth-First Search (BFS)
* Caching (FIFO Cache)
* Web Server Request Handling
* Messaging Systems
* Keyboard/Mouse Input Buffers

Let me know if you want to learn about:

* **Circular Queue**
* **Priority Queue**
* **Queue using Linked List**
* **Interview coding problems on Queues**